Electronic Transactions on Numerical Analysis.
Volume 10, 2000, pp. 41-55.
Copyright © 2000, Kent State University.
ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

A PARALLEL AMG FOR OVERLAPPING AND NON-OVERLAPPING DOMAIN DECOMPOSITION[*]

GUNDOLF HAASE[†]

Abstract. There exist several approaches for the parallel solving of huge systems of linear equations resulting from the finite element (f.e.) discretization of 2nd order elliptic pdes. On the other hand, there exists a great demand for Algebraic Multigrid solvers (AMG) which have as input only matrix and right hand side or, as a substitute, the appropriate information per element. In this paper we propose a general parallel AMG algorithm using overlapping or non-overlapping data decompositions.

Key words. algebraic multigrid, parallel algorithms, domain decomposition, iterative solvers.

AMS subject classifications. 65F10, 65N22, 65Y05.

1. Introduction. Without loss of generality we want to solve a second order pde with homogeneous Dirichlet boundary conditions in a domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ such that its weak formulation in Sobolev space $\mathbb{X}(\Omega)$ is represented by

$$\text{Find } u \in \mathbb{X}(\Omega) : \quad a(u,v) = \langle F, v \rangle \quad \forall v \in \mathbb{X}(\Omega) \tag{1.1}$$

with bilinear form $a(u,v) : \mathbb{X} \times \mathbb{X} \mapsto \mathbb{R}$ and duality product $\langle F, v \rangle : \mathbb{X}^* \times \mathbb{X} \mapsto \mathbb{R}$.

A discretization of the domain Ω results in a f.e. mesh τ_h consisting of elements δ^r and nodes $x^{[j]}$ with j from index set ω. The f.e. basis $\Phi = \left\{ \varphi^{[j]}(x) \, , \, j \in \omega \right\}$ has local support

$$\overline{\text{supp}}\, \varphi^{[i]} = \bigcup_{r:\, x^{[i]} \in \overline{\delta}^{(r)}} \overline{\delta}^{(r)} \tag{1.2}$$

and spans a f.e. space $\mathbb{V} = \text{span}\Phi$. Eq. (1.1) changes to

$$\text{Find } u \in \mathbb{V}(\Omega) \subseteq \mathbb{X} : \quad a(u,v) = \langle F, v \rangle \quad \forall v \in \mathbb{V}(\Omega) \ . \tag{1.3}$$

Now, the f.e. isomorphism

$$u(x) = \sum_{j \in \omega} u^{[j]} \cdot \varphi^{[j]}(x) = \Phi \cdot \underline{u} \ , \tag{1.4}$$

with $u^{[j]} = u(x^{[j]})$, guarantees the equivalence of (1.3) to the system of equations

$$K \cdot \underline{u} = \underline{f} \tag{1.5}$$

with

$$K^{[i,j]} = a(\varphi^{[j]}, \varphi^{[i]}) \tag{1.6}$$

$$f^{[i]} = \left\langle F, \varphi^{[i]} \right\rangle \qquad \forall i, j \in \omega \ . \tag{1.7}$$

Sections 2-3 introduce some definitions and data types used in the paper and generalize the parallelization strategy from [2, 4, 5, 6, 7] now, for overlapping domain decompositions.

[*] Received May 25, 1999. Accepted for publication February 1, 2000. Recommend by P. Vassilevski. This work has been partially supported by the Austrian Science Fund - "Fonds zur Förderung der wissenschaftlichen Forschung" - within the SFB F013 "Numerical and Symbolic Scientific Computing".

[†]Institute for Analysis and Computational Mathematics, Johannes Kepler University Linz, A-4040 Linz, Austria, (ghaase@numa.uni-linz.ac.at).

The new theoretical results on special matrix-times-vector operations are represented in Sec. 4 and impose requirements on the pattern of the matrices used in these operations. These abstract results are applied in Sec. 5 to derive a new parallel AMG algorithm, with slightly modified standard AMG components therein, for solving (1.5) in parallel.

2. Notation and examples.

2.1. Sets of subdomains and nodes. Denoting the discretization in subdomain s by $\tau_{s,h}$ then we can represent the discretized domain and its P subdomains by means of finite elements, i.e., $\overline{\Omega} = \bigcup_{\delta^{(r)}\in\tau_h} \overline{\delta}^{(r)}$ and $\overline{\Omega}_s = \bigcup_{\delta^{(r)}\in\tau_{s,h}} \overline{\delta}^{(r)}$, such that $\overline{\Omega} = \bigcup_{s=1}^{P} \overline{\Omega}_s$ holds. If $\Omega_s \cap \Omega_q = \emptyset$, $\forall q \neq s$, $s,q = [1,\dots,P]$, then this decomposition of Ω is called non-overlapping element decomposition. Otherwise it is an overlapping element decomposition.

Let us denote by M the binary matrix representing the correspondence between node and subdomain, namely

$$M_{is} := \begin{cases} 1 & \text{iff } x^{[i]} \in \overline{\Omega}_s \\ 0 & \text{iff } x^{[i]} \notin \overline{\Omega}_s \end{cases} . \tag{2.1}$$

The set of all those subdomains, a node $x^{[i]}$ belongs to, is denoted by

$$\sigma^{[i]} := \{s : M_{is} \neq 0\} , \tag{2.2}$$

which is equivalent to the statement

$$x^{[i]} \in \bigcap_{s\in\sigma^{[i]}} \overline{\Omega}_s . \tag{2.3}$$

We collect all nodes with the same σ-set of subdomains in the index set

$$\omega(\sigma) := \{i \in \omega : M_{is} \neq 0 \quad \forall s \in \sigma \text{ , and } M_{is} = 0 \quad \forall s \notin \sigma\} , \tag{2.4}$$

similar notations are introduced for those sets of nodes with appropriate subsets or supersets of σ:

$$\underline{\omega}(\sigma) := \{i \in \omega : M_{is} = 0 \quad \forall s \notin \sigma\} \tag{2.5}$$

$$\overline{\omega}(\sigma) := \{i \in \omega : M_{is} \neq 0 \quad \forall s \in \sigma\} . \tag{2.6}$$

REMARK 2.1. *Obviously,* $\omega(\emptyset) = \emptyset$ *and* $\sigma^{[i]} \neq \sigma^{[j]} \Leftrightarrow \omega(\sigma^{[i]}) \cap \omega(\sigma^{[j]}) = \emptyset$ *hold. The definitions (2.5) and (2.6) imply directly*

$$i \in \underline{\omega}(\sigma) \Leftrightarrow \sigma^{[i]} \subseteq \sigma \qquad \textit{and} \qquad i \in \overline{\omega}(\sigma) \Leftrightarrow \sigma^{[i]} \supseteq \sigma .$$

Using the above notation, we denote by

$$\omega_s = \{i \in \omega : M_{is} \neq 0\} \qquad (= \overline{\omega}(\{s\})) \tag{2.7}$$

the index set of subdomain Ω_s with $N_s := |\omega_s|$ nodes therein. Here and in the following, we denote by $|\cdot|$ the number of elements in a set.

REMARK 2.2. *Obviously, the statements*

$$i \in \omega_s \quad \Leftrightarrow \quad s \in \sigma^{[i]} \quad \Leftrightarrow \quad \{s\} \subseteq \sigma^{[i]} \tag{2.8}$$

$$j \notin \omega_s \quad \Leftrightarrow \quad s \notin \sigma^{[j]} \quad \Leftrightarrow \quad \{s\} \not\subseteq \sigma^{[j]} \tag{2.9}$$

are valid.

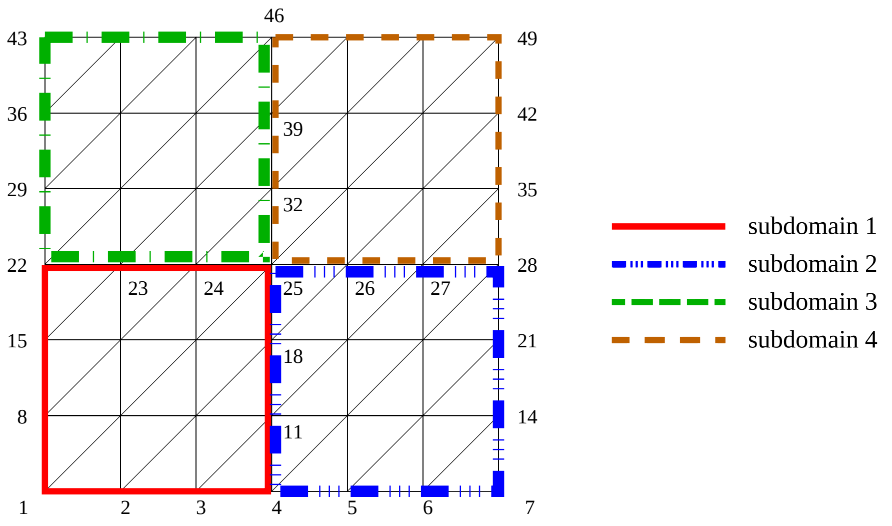

FIGURE 2.1. *Four non-overlapping subdomains with triangulation and row-wise numbering*

2.2. Examples. EXAMPLE 2.1 (Non-overlapping Elements). *As a first example, we divide the square into 4 non-overlapping squares and investigate the appropriate σ and $\omega(\sigma)$ sets. Let us represent all equal sets of subdomains by $\sigma^{[i]}$ with some fixed node i. Then we have the following adjacent sets :*

$$\begin{array}{llll} \sigma^{[1]} = \{1\}, & \sigma^{[7]} = \{2\}, & \sigma^{[43]} = \{3\}, & \sigma^{[49]} = \{4\}, \\ \sigma^{[4]} = \{1,2\}, & \sigma^{[22]} = \{1,3\}, & \sigma^{[28]} = \{2,4\}, & \sigma^{[46]} = \{3,4\}, \\ \sigma^{[25]} = \{1,2,3,4\}. & & & \end{array}$$

Some representative index sets are :

$$\begin{aligned} \omega(\sigma^{[1]}) = \omega(\{1\}) &= \{1,2,3,8,9,10,15,16,17\}, \\ \omega(\sigma^{[7]}) = \omega(\{2\}) &= \{5,6,7,12,13,14,19,20,21\}, \\ \omega(\sigma^{[4]}) = \omega(\{1,2\}) &= \{4,11,18\}, \\ \omega(\sigma^{[22]}) &= \{22,23,24\}, \\ \omega(\sigma^{[28]}) &= \{26,27,28\}, \\ \omega(\sigma^{[25]}) &= \{25\}. \end{aligned}$$

Some index sets derived by (2.5) and (2.6) :

$$\begin{array}{lll} \underline{\omega}(\sigma^{[1]}) & = \omega(\sigma^{[1]}), & \\ \overline{\omega}(\sigma^{[1]}) = \overline{\omega}(\{1\}) & = \omega(\sigma^{[1]}) \cup \omega(\sigma^{[4]}) \cup \omega(\sigma^{[22]}) \cup \omega(\sigma^{[25]}) & \\ & = \{\overline{1,4}, \overline{8,11}, \overline{15,18}, \overline{22,25}\}, & \\ \underline{\omega}(\sigma^{[4]}) & = \omega(\sigma^{[4]}) \cup \omega(\{1\}) \cup \omega(\{2\}) & = \{\overline{1,21}\}, \\ \overline{\omega}(\sigma^{[4]}) & = \omega(\sigma^{[4]}) \cup \omega(\sigma^{[25]}) & = \{4,11,18,25\}, \\ \underline{\omega}(\sigma^{[25]}) & & = \{\overline{1,49}\}, \\ \overline{\omega}(\sigma^{[25]}) & = \omega(\sigma^{[25]}) & = \{25\}. \end{array}$$

EXAMPLE 2.2 (Overlapping Elements). *Here, we divide the square into 4 overlapping squares with an overlap of 2 mesh sizes.*

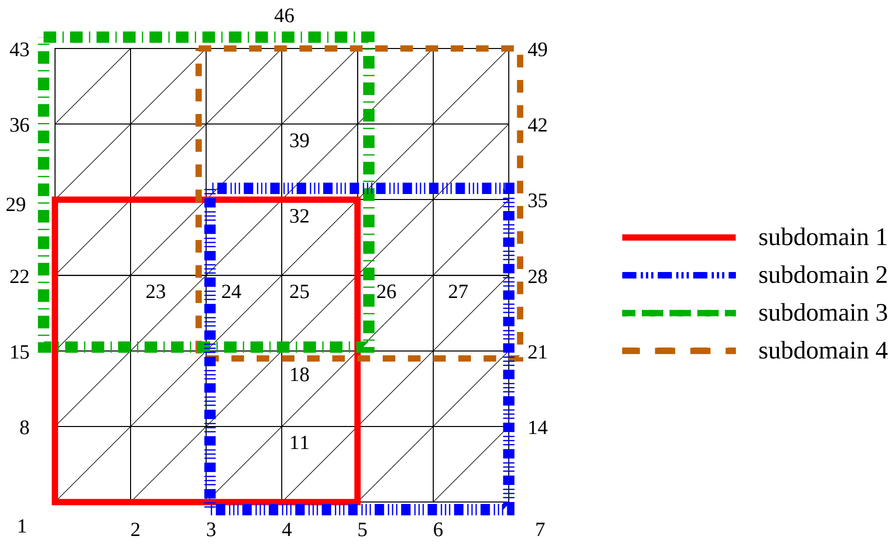

FIGURE 2.2. *Four overlapping subdomains with triangulation and row-wise numbering*

The subdomain sets $\sigma^{[i]}$ are represented in the same way as in Ex. 2.1 and result in the identical sets. On the other hand, we achieve different index sets, e.g.,

$$
\begin{aligned}
\omega(\sigma^{[1]}) &= \{1,2,8,9\}, & \omega(\sigma^{[7]}) &= \{6,7,13,14\},\\
\omega(\sigma^{[43]}) &= \{36,37,43,44\}, & \omega(\sigma^{[49]}) &= \{41,42,48,49\},\\
\omega(\sigma^{[4]}) &= \{3,4,5,10,11,12\}, & \omega(\sigma^{[22]}) &= \{15,16,22,23,29,30\},\\
\omega(\sigma^{[28]}) &= \{20,21,27,28,34,35\}, & \omega(\sigma^{[46]}) &= \{38,39,40,45,46,47\},\\
\omega(\sigma^{[25]}) &= \{17,18,19,24,25,26,31,32,33\}.
\end{aligned}
$$

We achieve the following derived index sets:

$$
\begin{aligned}
\underline{\omega}(\sigma^{[1]}) &= \omega(\sigma^{[1]}),\\
\overline{\omega}(\sigma^{[1]}) &= \{\overline{1,5},\overline{8,12},\overline{15,19},\overline{22,25},\overline{29,33}\},\\
\underline{\omega}(\sigma^{[4]}) &= \{\overline{1,14}\},\\
\overline{\omega}(\sigma^{[4]}) &= \{3,4,5,10,11,12,17,18,19,24,25,26,31,32,33\},\\
\underline{\omega}(\sigma^{[25]}) &= \{\overline{1,49}\},\\
\overline{\omega}(\sigma^{[25]}) &= \omega(\sigma^{[25]}).
\end{aligned}
$$

2.3. The coincidence matrix. To represent the mapping of a vector $\underline{u} \in \mathbb{R}^{\omega}$ in global numbering onto a local vector $\underline{u}_s \in \mathbb{R}^{\omega_s}$ in subdomain $\overline{\Omega}_s$ ($s = [1,\dots,P]$), we introduce coincidence matrices A_s of dimension $N_s \times N$ with entries

$$A_s^{[i,j]} := \begin{cases} 1 & \text{if} \quad j = \text{global number of } i \\ 0 & \text{else} \end{cases} \qquad \forall i \in \omega_s\,,\ \forall j \in \omega\ . \tag{2.10}$$

The transpose A_s^T of these binary matrices $A_s : \mathbb{R}^{\omega} \mapsto \mathbb{R}^{\omega_s}$ maps a local vector back to the global one.

LEMMA 2.1. *By (2.7) and (2.10) we achieve for all $s = [1, \ldots, P]$ the arithmetic relations:*

$$A_s^T \cdot A_s{}_{N\times N} = \operatorname*{diag}_{i\in\omega} \begin{cases} 1 & \text{iff } i \in \omega_s \\ 0 & \text{iff } i \notin \omega_s \end{cases} \tag{2.11a}$$

$$A_s \cdot A_s^T{}_{N_s\times N_s} = \operatorname*{diag}_{i\in\omega_s}\{1\} \tag{2.11b}$$

3. Vector and matrix types.

3.1. Vector types. In all domain decomposition methods we have a certain subset of nodes which belong to more than one subdomain, e.g., nodes $22, 25$ in Fig. 2.1 and nodes $21, 22, 24, 25$, in Fig. 2.2.

We store the data of a vector component $u^{[i]}$ in each subdomain Ω_s that component belongs to, i.e., $s \in \sigma^{[i]}$. There are (at least) two opportunities to store those components and at the end of the vector.

DEFINITION 3.1 (accumulated vector). *A vector $\underline{\mathfrak{u}}_s$ ($\underline{\mathfrak{u}}$) is called an accumulated vector if each vector component $\mathfrak{u}^{[i]}$ is stored in all subdomains Ω_s, $s \in \sigma^{[i]}$ with its full value. The local vectors $\underline{\mathfrak{u}}_s$ can be represented as*

$$\underline{\mathfrak{u}}_s := A_s \cdot \underline{\mathfrak{u}}\ . \tag{3.1}$$

DEFINITION 3.2 (distributed vector). *A vector $\underline{\mathsf{r}}$ is called a distributed vector if it is decomposed into local vectors $\underline{\mathsf{r}}_s$ such that*

$$\underline{\mathsf{r}} = \sum_{s=1}^{P} A_s^T \cdot \underline{\mathsf{r}}_s \tag{3.2}$$

holds, i.e., all subdomains Ω_s, $s \in \sigma^{[i]}$ store only $\underline{\mathsf{r}}_s$ and possess a portion of the full vector value $\mathsf{r}^{[i]}$ which can be determined only by summing in (3.2).

The conversion of a distributed vector $\underline{\mathsf{v}}$ into an accumulated vector $\underline{\mathfrak{w}}$ can be done by performing the sum in (3.2) followed by restriction (3.1), i.e.,

$$\underline{\mathfrak{w}} \leftarrow \underline{\mathsf{v}} \quad : \quad \underline{\mathfrak{w}}_s := A_s \cdot \underline{\mathfrak{w}} = A_s \cdot \sum_{s=1}^{P} A_s^T \cdot \underline{\mathsf{v}}_s\ . \tag{3.3}$$

The conversion in the other direction is not unique - we prefer an equal weighted distribution of the accumulated vector. A matrix weighted distribution is also feasible in case of discontinuous coefficients. The weights are chosen such that re-conversion (3.3) will result in the original vector :

$$\underline{\mathsf{v}} \leftarrow \underline{\mathfrak{w}} \quad : \quad \underline{\mathsf{v}}_s := (R_s)^{-1} \cdot \underline{\mathfrak{w}}_s\ , \tag{3.4}$$

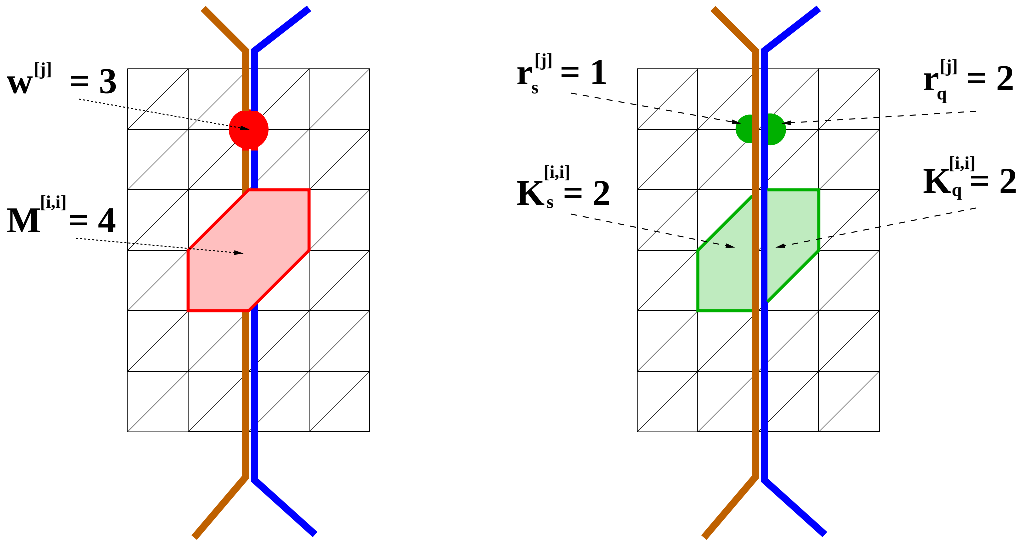

FIGURE 3.1. *Illustration for accumulated and distributed vectors and matrices.*

with $R_s = \operatorname*{diag}_{i\in\omega_s}\{|\sigma^{[i]}|\}$, i.e., $R_s^{[i,i]}$ stores the number of subdomains node i belongs to.

REMARK 3.1. *It can be seen easily by (2.11a) that*

$$R := \operatorname*{diag}_{i\in\omega}\{|\sigma^{[i]}|\} = \sum_{s=1}^{P} A_s^T \cdot A_s \quad . \tag{3.5}$$

3.2. Matrix types. The matrix defined by the bilinear form in (1.6) can also be stored in two ways. With respect to an element-wise domain decomposition, we can store the f.e. matrix either as an accumulated or distributed matrix.

DEFINITION 3.3 (accumulated matrix). *A matrix $\mathfrak{M}$ is called accumulated if its local restrictions $\mathfrak{M}_s$ possess the full entries of it and we can write*

$$\mathfrak{M}_s := A_s \cdot \mathfrak{M} \cdot A_s^T \quad . \tag{3.6}$$

DEFINITION 3.4 (distributed matrix). *We call a matrix K distributed if we have locally stored matrices K_s such that*

$$\mathsf{K} := \sum_{s=1}^{P} A_s^T \cdot \mathsf{K}_s \cdot A_s \quad . \tag{3.7}$$

holds, i.e., each subdomain $\overline{\Omega}_s$ stores only a part of its full values.
Similar to conversion of vectors, the construction of a distributed matrix from an accumulated one is not unique.

3.3. Construction of distributed matrices. The construction of distributed matrices for a non-overlapping element decomposition (see Ex. 2.1) is well known. Here, we generalize that approach for arbitrary overlapping subdomains.

In the f.e. context, a matrix entry $K^{[i,j]}$ is obtained from the sum over all bilinear forms on the elements

$$K^{[i,j]} \;:=\; a_\Omega(\varphi^{[j]},\varphi^{[i]}) \;=\; \sum_{\delta^{(r)}\subseteq\Omega} a_{\delta^{(r)}}(\varphi^{[j]},\varphi^{[i]}) \;\;, \tag{3.8}$$

with basis functions $\varphi^{[i]}(x)$, $i\in\omega$ and the bilinear form from Sec. 1.

A decomposition of $\overline{\Omega} = \bigcup_{s=1}^{P}\overline{\Omega}_s$ results

for the bilinear form $a(\cdot,\cdot)$ in a weighted summation with appropriate weights $W^{(r)}$ over all subdomains and all their elements.

$$\begin{aligned} a_\Omega(\varphi^{[j]},\varphi^{[i]}) &= \sum_{s=1}^{P}\sum_{\delta^{(r)}\subseteq\Omega_s}\frac{1}{W^{(r)}}a_{\delta^{(r)}}(\varphi^{[j]},\varphi^{[i]}) \\ &= \sum_{\delta^{(r)}\subseteq\Omega} a_{\delta^{(r)}}(\varphi^{[j]},\varphi^{[i]})\cdot\left(\frac{1}{W^{(r)}}\sum_{s:\,\delta^{(r)}\subseteq\Omega_s} 1\right). \end{aligned} \tag{3.9}$$

The comparison of (3.8) with (3.9) gives us directly the definition for the weights per element.

THEOREM 3.5 (Construction principle for distributed matrices). *In the multi-domain approach with $\overline{\Omega} = \bigcup_{s=1}^{P}\overline{\Omega}_s$, a distributed f.e. matrix*

$$\mathsf{K} \;:=\; \sum_{s=1}^{P} A_s^T\mathsf{K}_s A_s \;\;. \tag{3.7}$$

has to be calculated in each subdomain by

$$\mathsf{K}_s^{[i,j]} \;=\; \sum_{\delta^{(r)}\subseteq\Omega_s}\frac{1}{W^{(r)}}\underbrace{a_{\delta^{(r)}}(\varphi^{[j]},\varphi^{[i]})}_{K_{\delta^{(r)}}^{[i,j]}} \qquad \forall i,j\in\omega_s, \tag{3.10}$$

with the weights for each element of the mesh

$$W^{(r)} \;:=\; \sum_{s:\,\delta^{(r)}\subseteq\Omega_s} 1 \;\;. \tag{3.11}$$

Proof. The proof was given from Eq. (3.8) to (3.9). □

REMARK 3.2. *Eq. (3.10) describes the local accumulation of weighted element matrices. The weight $W^{(r)}$ in (3.11) is the number of subdomains Ω_s an element $\delta^{(r)}$ belongs to, i.e.,*

$$W^{(r)} \;=\; \left|\bigcap_{x_i\in\overline{\delta}^{(r)}}\sigma^{[i]}\right| \;. \tag{3.12}$$

- The non-overlapping elements in Fig. 2.1 imply $W^{(r)}\equiv 1$ for all elements, so that no weighting is necessary and K_s is the local accumulated f.e. matrix.
- The weights for the overlapping elements in Fig. 2.2 are 1, 2 or 4.

ETNA
Kent State University
etna@mcs.kent.edu

4. Matrix-times-vector operations.

4.1. Abstract results. It can be easily seen by Eq. (2.10), (3.1)-(3.3) and (3.7) that

$$\mathsf{K}\cdot\underline{\mathfrak{w}} = \sum_{s=1}^{P} A_s^T \mathsf{K}_s A_s \cdot \underline{\mathfrak{w}} \;=\; \sum_{s=1}^{P} A_s^T \mathsf{K}_s \cdot \underline{\mathfrak{w}}_s \;=\; \sum_{s=1}^{P} A_s^T \underline{\mathsf{v}}_s \;=\; \underline{\mathsf{v}} \tag{4.1}$$

can be performed in parallel without communication and results in a distributed vector $\underline{\mathsf{v}}$ with $\underline{\mathsf{v}}_s = \mathsf{K}_s \cdot \underline{\mathfrak{w}}_s$. A multiplication with a distributed vector instead of $\underline{\mathfrak{w}}$ requires conversion to an accumulated one through (3.3).

The multiplication of an accumulated matrix $\mathfrak{M}$ with a vector requires more detailed investigations. It is not necessary that Matrix $\mathfrak{M}$ has to be a f.e. matrix. The following two theorems will present conditions for admissible matrix patterns for certain classes of multiplications.

THEOREM 4.1. *The multiplication of an accumulated matrix with an accumulated vector results in a vector of the same type*

$$\underline{\mathfrak{w}} \;=\; \mathfrak{M}\cdot\underline{\mathfrak{u}} \tag{4.2}$$

and can be performed without communication in parallel, i.e.,

$$\underline{\mathfrak{w}}_s \;=\; \mathfrak{M}_s\cdot\underline{\mathfrak{u}}_s \qquad \forall s=[1,\dots,P]\ , \tag{4.3}$$

if and only if

$$\forall i,j\in\omega\ : \qquad \sigma^{[i]}\not\subseteq\sigma^{[j]} \implies \mathfrak{M}^{[i,j]}=0 \tag{4.4}$$

is true.

Proof. We start with definition (3.1) of an accumulated vector

$$\underline{\mathfrak{w}}_s = A_s\underline{\mathfrak{w}} \qquad\qquad \forall s=[1,\dots,P]\ ,$$

rewrite it by preliminaries (4.3), (4.2) and definition (3.6) of an accumulated matrix

$$\begin{aligned}
&\mathfrak{M}_s\underline{\mathfrak{u}}_s = A_s\mathfrak{M}\underline{\mathfrak{u}} && \forall s=[1,\dots,P]\\
\iff\quad &\left(A_s\mathfrak{M}A_s^T\cdot\underline{\mathfrak{u}}_s\right)^{[i]} = \left(A_s\mathfrak{M}\cdot\underline{\mathfrak{u}}\right)^{[i]} && \forall s=[1,\dots,P],\quad \forall i\in\omega_s\ ,
\end{aligned}$$

and investigate the results of the multiplications on both sides component-wise in global numbering

$$\sum_{j\in\omega_s}\mathfrak{M}^{[i,j]}\mathfrak{u}^{[j]} = \sum_{j\in\omega}\mathfrak{M}^{[i,j]}\mathfrak{u}^{[j]} \qquad \forall s=[1,\dots,P],\quad \forall i\in\omega_s\ .$$

The statement of the last line is true if and only if the following is true

$$\begin{aligned}
&\forall s=[1,\dots,P],\quad \forall i\in\omega_s\ : & j\notin\omega_s &\implies \mathfrak{M}^{[i,j]}=0\\
\overset{(2.8),(2.9)}{\iff}\quad &\forall s=[1,\dots,P]\ : & \{s\}\in\sigma^{[i]}\wedge\{s\}\notin\sigma^{[j]} &\implies \mathfrak{M}^{[i,j]}=0\\
\iff\quad & & \sigma^{[i]}\not\subseteq\sigma^{[j]} &\implies \mathfrak{M}^{[i,j]}=0
\end{aligned}$$

□

Theorem 4.1 says that operation (4.2) can be performed locally (4.3) without communication only for special matrices $\mathfrak{M}$. Condition (4.4) on the matrix pattern can be easily verified, e.g. in Fig. 2.1, the matrix entries $\mathfrak{M}^{[25,24]} = \mathfrak{M}^{[4,5]}$ have to be zero but not $\mathfrak{M}^{[24,25]}$, $\mathfrak{M}^{[5,4]}$. Note, that $\mathfrak{M}^{[18,26]}$ as well as $\mathfrak{M}^{[26,18]}$ have to be zero because there is no subset relation at all between $\sigma^{[18]} = \{1,2\}$ and $\sigma^{[26]} = \{2,4\}$.

THEOREM 4.2. *The multiplication of an accumulated matrix with a distributed vector results in a vector of the same type*

$$\underline{\mathsf{r}} = \mathfrak{M} \cdot \underline{\mathsf{v}} \tag{4.5}$$

and can be performed without communication in parallel, i.e.,

$$\underline{\mathsf{r}}_s = \mathfrak{M}_s \cdot \underline{\mathsf{v}}_s \qquad \forall s = [1, \ldots, P] \ , \tag{4.6}$$

if and only if

$$\forall i, j \in \omega : \qquad \sigma^{[i]} \not\supseteq \sigma^{[j]} \implies \mathfrak{M}^{[i,j]} = 0 \tag{4.7}$$

is true.

Proof. We start with definition (3.2) of a distributed vector

$$\sum_{s=1}^{P} A_s^T \underline{\mathsf{r}}_s = \underline{\mathsf{r}} \ ,$$

reformulate it by preliminaries (4.6), (4.5) and apply definitions for accumulated matrix and distributed vector on $\mathfrak{M}_s$ and $\underline{\mathsf{v}}$

$$\sum_{s=1}^{P} A_s^T A_s \mathfrak{M}_s A_s^T \cdot \underline{\mathsf{v}}_s \overset{(3.6)}{=} \sum_{s=1}^{P} A_s^T \mathfrak{M}_s \cdot \underline{\mathsf{v}}_s = \mathfrak{M} \cdot \underline{\mathsf{v}} \overset{(3.2)}{=} \mathfrak{M} \cdot \sum_{s=1}^{P} A_s^T \underline{\mathsf{v}}_s \ .$$

Applying (2.11a) in a component-wise investigation using global numbering results in

$$\begin{pmatrix} \sum\limits_{s=1}^{P} \sum\limits_{j \in \omega_s} \mathfrak{M}^{[i,j]} \cdot \mathsf{v}_s^{[j]} & , & i \in \omega_s \\ 0 & , & i \notin \omega_s \end{pmatrix} = \begin{pmatrix} \sum\limits_{j \in \omega} \mathfrak{M}^{[i,j]} \cdot \sum\limits_{s=1}^{P} \mathsf{v}_s^{[j]} & , & i \in \omega \end{pmatrix} \ ,$$

which is equivalent to

$$\begin{pmatrix} \sum\limits_{j \in \omega} \sum\limits_{s \in \sigma^{[j]}} \mathfrak{M}^{[i,j]} \cdot \mathsf{v}_s^{[j]} & , & i \in \omega_s \\ 0 & , & i \notin \omega_s \end{pmatrix} = \begin{pmatrix} \sum\limits_{j \in \omega} \sum\limits_{s=1}^{P} \mathfrak{M}^{[i,j]} \cdot \mathsf{v}_s^{[j]} & , & i \in \omega \end{pmatrix} \ .$$

The statement of the last line is true if and only if

$$\begin{aligned} s \in \sigma^{[j]} \wedge i \notin \omega_s &\implies \mathfrak{M}^{[i,j]} = 0 \\ \{s\} \subseteq \sigma^{[j]} \wedge \{s\} \not\subseteq \sigma^{[i]} &\implies \mathfrak{M}^{[i,j]} = 0 \\ \sigma^{[j]} \not\subseteq \sigma^{[i]} &\implies \mathfrak{M}^{[i,j]} = 0 \end{aligned}$$

□

Condition (4.7) on the matrix pattern can be easily verified, e.g. in Fig. 2.1, the matrix entries $\mathfrak{M}^{[24,25]}$, $\mathfrak{M}^{[5,4]}$. have to be zero but not $\mathfrak{M}^{[25,24]}$ and $\mathfrak{M}^{[4,5]}$. Again, $\mathfrak{M}^{[18,26]}$ as well as $\mathfrak{M}^{[26,18]}$ have to be zero.

REMARK 4.1. *Intermediate steps in the proofs of theorems 4.1 and 4.2 lead directly to equivalent formulations for (4.4)*

$$\mathfrak{M} A_s^T A_s = \mathfrak{M} \qquad \forall s = [1, \ldots, P] \tag{4.8}$$

and for (4.7)

$$A_s^T A_s \mathfrak{M} = \mathfrak{M} \qquad \forall s = [1, \dots, P] \ . \tag{4.9}$$

Obviously, we can conclude that if a matrix $\mathfrak{M}$ fulfills (4.4) then $\mathfrak{M}^T$ fulfills also (4.7). Eq. (4.8) and (4.9) have been derived for non-overlapping element domain decompositions by Groh [1] as a condition to apply (4.2) and (4.5) in parallel.

REMARK 4.2. *The sets of subdomains induce a block structure of the unknowns in that way that all nodes with the same σ-set, e.g. σ_k, build one block (see subsection 4.2 for an example). Now, it follows directly from theorems 4.1 and 4.2 that an accumulated matrix $\mathfrak{M}$ can be applied to (4.2) <u>and</u> (4.5) iff $\mathfrak{M}$ possesses a block diagonal structure induced by the described block structure of nodes. We can write*

$$\mathfrak{M} \ = \ \text{blockdiag}\left\{\mathfrak{M}_k\right\} \qquad \textit{with} \qquad \mathfrak{M}_k \ = \ \left\{\mathfrak{M}^{[i,j]}\right\}_{i,j\in\omega(\sigma_k)} \ . \tag{4.10}$$

REMARK 4.3. *If $\mathfrak{M}$ fulfills (4.10) and all $\mathfrak{M}_k^{-1}$ do exist then also the inverse $\mathfrak{M}^{-1}$ fulfills the proper block condition and can be applied in parallel on (4.2) and (4.5).*
The idea of the following theorem originates from the observation that the result of (4.2) can be used as input for operation (4.1), that this result has exactly the right vector type to serve as input for (4.5) and that all those operations can be performed fully in parallel. matrices

THEOREM 4.3. *If the accumulated matrix $\mathfrak{P}$ fulfills pattern condition (4.4) then the matrix multiplications $\mathfrak{P}^T \cdot \mathsf{K} \cdot \mathfrak{P}$ result in a distributed matrix K^H which can be calculated locally in parallel by simply computing $\mathsf{K}_s^H = \mathfrak{P}_s^T \cdot \mathsf{K}_s \cdot \mathfrak{P}_s$, i.e.,*

$$\mathsf{K}^H \ = \ \sum_{s=1}^{P} A_s^T \mathsf{K}_s^H A_s \ = \ \sum_{s=1}^{P} A_s^T \left(\mathfrak{P}_s^T \cdot \mathsf{K}_s \cdot \mathfrak{P}_s\right) A_s \ = \ \mathfrak{P}^T \cdot \mathsf{K} \cdot \mathfrak{P} \ . \tag{4.11}$$

Proof. We substitute K in the right expression of (4.11)
by its definition (3.7)

$$\mathfrak{P}^T \cdot \mathsf{K} \cdot \mathfrak{P} = \mathfrak{P}^T \cdot \sum_{s=1}^{P} A_s^T \mathsf{K}_s A_s \cdot \mathfrak{P} \ = \ \sum_{s=1}^{P} \mathfrak{P}^T A_s^T \mathsf{K}_s A_s \mathfrak{P}$$

and apply the equivalent pattern conditions (4.8) and (4.9) for all subdomains

$$= \sum_{s=1}^{P} A_s^T A_s \mathfrak{P}^T A_s^T \mathsf{K}_s A_s \mathfrak{P} A_s^T A_s \ .$$

Using definition (3.6) for $\mathfrak{P}_s$ yields finally

$$= \sum_{s=1}^{P} A_s^T \underbrace{\left(\mathfrak{P}_s^T \cdot \mathsf{K}_s \cdot \mathfrak{P}_s\right)}_{=:\mathsf{K}_s^H} A_s \ = \ \sum_{s=1}^{P} A_s^T \mathsf{K}_s^H A_s \ .$$

□

4.2. Application of theorems 4.1 and 4.2. We will apply the results of theorems 4.1 and 4.2 on our Ex. 2.1 - 2.2.

There are 9 different sets of subdomains in Ex. 2.1. These sets will be arranged with respect to the number of subdomains therein :

$$\begin{array}{llll} \sigma_V = \sigma^{[25]}, & & & \\ \sigma_{E_1} = \sigma^{[4]}, & \sigma_{E_2} = \sigma^{[22]}, & \sigma_{E_3} = \sigma^{[28]}, & \sigma_{E_4} = \sigma^{[46]}, \\ \sigma_{I_1} = \sigma^{[1]}, & \sigma_{I_2} = \sigma^{[7]}, & \sigma_{I_3} = \sigma^{[43]}, & \sigma_{I_4} = \sigma^{[49]} \ . \end{array}$$

The subscript V indicates vertex nodes, subscripts E and I denote

edge and interior nodes of the subdomains. The block structure above implies an appropriate block structure in vectors and matrices. The admissible matrix pattern wrt. (4.4) is

(4.12)

$$\mathfrak{M} = \begin{pmatrix} \mathfrak{M}_V & & & & & & & & \\ \mathfrak{M}_{E_1V} & \mathfrak{M}_{E_1} & & & & & & & \\ \mathfrak{M}_{E_2V} & 0 & \mathfrak{M}_{E_2} & & & & & & \\ \mathfrak{M}_{E_3V} & 0 & 0 & \mathfrak{M}_{E_3} & & & 0 & & \\ \mathfrak{M}_{E_4V} & 0 & 0 & 0 & \mathfrak{M}_{E_4} & & & & \\ \mathfrak{M}_{I_1V} & \mathfrak{M}_{I_1E_1} & \mathfrak{M}_{I_1E_2} & 0 & 0 & \mathfrak{M}_{I_1} & & & \\ \mathfrak{M}_{I_2V} & \mathfrak{M}_{I_2E_1} & 0 & \mathfrak{M}_{I_2E_3} & 0 & 0 & \mathfrak{M}_{I_2} & & \\ \mathfrak{M}_{I_3V} & 0 & \mathfrak{M}_{I_3E_2} & 0 & \mathfrak{M}_{I_3E_4} & 0 & 0 & \mathfrak{M}_{I_3} & \\ \mathfrak{M}_{I_4V} & 0 & 0 & \mathfrak{M}_{I_4E_3} & \mathfrak{M}_{I_4E_4} & 0 & 0 & 0 & \mathfrak{M}_{I_4} \end{pmatrix} .$$

This matrix can be also expressed in the form

$$\mathfrak{M} = \begin{pmatrix} \mathfrak{M}_V & 0 & 0 \\ \mathfrak{M}_{EV} & \mathfrak{M}_E & 0 \\ \mathfrak{M}_{IV} & \mathfrak{M}_{IE} & \mathfrak{M}_I \end{pmatrix} , \tag{4.13}$$

requiring the appropriate block structures of the submatrices from (4.12). Especially, we have to require that $\mathfrak{M}_V$, $\mathfrak{M}_E$ and $\mathfrak{M}_I$ are block diagonal matrices (Fig. 2.1: $\mathfrak{M}_E^{[18,26]} = \mathfrak{M}_E^{[26,18]} \overset{!}{=} 0$!),

from which follows by remark 4.3 that we can apply also the inverse of them.

The previous investigations are completely valid for Ex. 2.2! The only difference consists in the different sets of nodes grouped within the blocks.

5. A parallel AMG.

5.1. Parallel MG. With the data structures and methods introduced in the previous sections

we can formulate a parallel MG algorithm; see also [5]. The components in Alg.1 have to be chosen such that the following requirements are fulfilled.

- The interpolation of the correction $\underline{\mathfrak{w}} := \mathfrak{P}\underline{\mathfrak{w}}^H$ needs an accumulated interpolation matrix with an *admissible matrix pattern*; see theorem 4.1. If $\mathfrak{P}$ possesses such a pattern then also $\mathfrak{P}^T$ can be applied in the defect restriction.
  If all refined elements belong to the same subdomain as their appropriate coarse element, then the pattern condition on the interpolation is fulfilled automatically.
- We assume that we have a parallel smoother; e.g. see [2].
- The coarse matrix K^H has to be again a distributed matrix to ensure the continuation of the recursive algorithm.
- The matrix K^H has to preserve the null space of K.

ETNA
Kent State University
etna@mcs.kent.edu

Algorithm 1 Parallel multigrid PMG$(\mathsf{K}, \underline{\mathfrak{u}}, \underline{\mathfrak{f}}, \ell)$

```
if $\ell == 1$ then
    solve $\sum_{s=1}^{P} A_s^T \mathsf{K} A_s \cdot \underline{\mathfrak{u}} \; = \; \underline{\mathfrak{f}}$
else
    $\underline{\widetilde{\mathfrak{u}}} \leftarrow$ SMOOTH$(\mathsf{K}, \underline{\mathfrak{u}}, \underline{\mathfrak{f}}, \nu)$
    $\underline{\mathfrak{d}} \leftarrow \underline{\mathfrak{f}} - \mathsf{K} \cdot \underline{\mathfrak{u}}$
    $\underline{\mathfrak{d}}^H \leftarrow \mathfrak{P}^T \cdot \underline{\mathfrak{d}}$
    $\underline{\mathfrak{w}}^H \leftarrow 0$
    PMG$(\mathsf{K}^H, \underline{\mathfrak{w}}^H, \underline{\mathfrak{d}}^H, \ell - 1)$
    $\underline{\mathfrak{w}} \leftarrow \mathfrak{P} \cdot \underline{\mathfrak{w}}^H$
    $\underline{\widehat{\mathfrak{u}}} \leftarrow \underline{\widetilde{\mathfrak{u}}} + \underline{\mathfrak{w}}$
    $\underline{\mathfrak{u}} \leftarrow$ SMOOTH$(\mathsf{K}, \underline{\widehat{\mathfrak{u}}}, \underline{\mathfrak{f}}, \nu)$
end if
```

5.2. Sequential AMG. Each AMG algorithm consists of the following parts:

1. Coarsening, i.e., $\omega = \omega_C \cup \omega_F$.
2. Determining the interpolation weights $P = \{\alpha_{ij}\}_{i \in \omega,\, j \in \omega_C}$.
   Typically, $\alpha_{i,j} = \delta_{ij}, \forall i, j \in \omega_C$ holds.
3. Calculate the coarse matrix

$$K^H \; = \; P^T \cdot K \cdot P \; . \tag{5.1}$$

4. Apply some multigrid procedure using the components determined in 1.-3.

5.2.1. Sequential coarsening. We can use any coarsening strategy for our investigations. Therefore, we present only the simple coarsening instead of the classical approach [8]. We only have to adapt the sequential algorithm such that it works on an arbitrary given set of nodes, i.e., a subset of all nodes.

Algorithm 2 Simple coarsening COARSE$(K, \omega, \omega_C, \omega_F)$

```
$\omega_t \leftarrow \omega$
while $\omega_t \neq \emptyset$ do
    pick some $i \in \omega_t$ (strong connections in $K|_{\omega}$)
    $\omega_C \leftarrow \omega_C \cup \{i\}$
    determine neighbors of $i \longrightarrow U_i$
    $\omega_F \leftarrow \omega_F \cup (U_i \cap \omega_t)$
    $\omega_t \leftarrow \omega_t \setminus (\{i\} \cup U_i \cap \omega_t)$
end while
```

5.2.2. Determination of interpolation weights. Here, we restrict the calculation of interpolation weights α_{ij}

on a given index set. Again, we can use an arbitrary routine with adaption for subsets of nodes.
A correct chosen matrix dependent interpolation is equivalent to the energy minimization technique for determining the interpolation weights. In case of the Laplacian, the condition $\sum_{j \in \omega} \alpha_{i,j} = 1\,,\ \forall i \in \omega$, guarantees that the null space of that operator can be represented on coarser grids.

Algorithm 3 Calculation of interpolation weights WEIGHTS$(K, \omega_C, \omega_F, \alpha)$

```
for all $i \in \omega_F$ do
  determine neighbors of $i \longrightarrow U_i$
  for all $j \in U_i \cap \omega_C$ do
    calculate $\alpha_{ij}$
  end for
end for
```

5.2.3. Calculation of coarse matrix. The classical Galerkin approach

$$\left(K^H\right)^{[ij]} = \sum_{k\in\omega}\sum_{l\in\omega} \alpha_{ki} \cdot K^{[kl]} \cdot \alpha_{lj} \qquad \forall i,j \in \omega_C \tag{5.2}$$

leads directly to a standard routine for calculating the coarse grid matrix. Here, no adaption is necessary.

Algorithm 4 Calculation of coarse matrix GALERKIN$(K^H, K, \omega_C, \omega_F, \alpha)$

```
Require: $K^H = 0$ before first call
  for all $k \in \omega_F \cup \omega_C$ do
    determine fine and coarse neighbors of $k \longrightarrow F^k, C^k$
    for all $l \in F^k \cup C^k \cup \{k\}$ do
      $T \leftarrow C^k \cup C^l \cup (\{l\} \cap \omega_C)$
      for all $i \in T$ do
        for all $j \in T$ do
          $\left(K^H\right)^{[ij]} \leftarrow \left(K^H\right)^{[ij]} + \alpha_{ki} \cdot (K)^{[kl]} \cdot \alpha_{lj}$
        end for
      end for
    end for
  end for
```

This Alg. 4 can by dramatically accelerated by use of matrix structures.

5.3. The parallel coarsening. The control of the admissible pattern for interpolation and restriction (4.4,4.7) will be realized by appropriate subsets of nodes as input parameters for Alg. 2–4. Controlling these subsets is the main task of Alg. 5.

Algorithm 5 Parallel coarsening SIGMACOARSE$(K, \sigma, \omega, \omega_C, \omega_F, \alpha)$

```
if $\omega == \emptyset$ then
  return
else
  $\omega_t \leftarrow \omega(\sigma)$
  COARSE$(K, \omega_t, \omega_C, \omega_F)$
  WEIGHTS$(K, \omega_C \cap \overline{\omega}(\sigma), \omega_F \cap \omega_t, \alpha)$
  $\omega \leftarrow \omega \setminus \omega_t$
  for all $\hat{\sigma} \subset \sigma$ such that $\nexists \tau \subset \sigma \,:\, \hat{\sigma} \subset \tau$ do
    SIGMACOARSE$(K, \hat{\sigma}, \omega, \omega_C, \omega_F, \alpha)$
  end for
end if
```

ETNA
Kent State University
etna@mcs.kent.edu

The initial call of the coarsening is SIGMACOARSE($\{1,\dots,P\},\omega^h,\emptyset,\emptyset$) on all processors. This means in Ex. 2.1 on page 43 that for this first call $\omega_t = \{25\}$, so that the only vertex node 25 will be automatically a coarse node. The biggest subsets $\hat{\sigma}$ of $\{1,\dots,P\}$ with non-empty $\omega(\hat{\sigma})$ are $\{1,2\}$, $\{1,3\}$, $\{2,4\}$, $\{3,4\}$ which continue the coarsening on the four interface edges. These steps are followed by coarsening in the interior of subdomains.
As a result, e.g, interpolation from vertex node to edge node or interior node may happen but not vice versa.

- The statement $\omega := \omega \setminus \omega_t$ blocks redundant calculations by reducing the set of available nodes.
- The recursive algorithm comes to an end if σ consists of only one element.
- The special parameters in the call
  WEIGHTS($K,\omega_C\cap\overline{\omega}(\sigma),\omega_F\cap\omega_t,\alpha$)
  guarantee an admissible matrix pattern in $\mathfrak{P}$, because

$$\left.\begin{array}{rcl}\omega_C\cap\overline{\omega}(\sigma) & = & \{j\in\omega_C\ :\ \sigma^{[j]}\supseteq\sigma\}\\ \omega_F\cap\omega(\sigma) & = & \{i\in\omega_F\ :\ \sigma^{[i]}=\sigma\}\end{array}\right\}\Longrightarrow\ \sigma^{[i]}\subseteq\sigma^{[j]}\ ,\tag{5.3}$$

  i.e., by (4.4) that only those α_{ij} are calculated in WEIGHTS which fulfill the pattern condition on the interpolation $\mathfrak{P}$.
- The usage of blocking coarsening requires some semaphore structure to synchronize the work in the branches, i.e., which processor performs the coarsening on a set of intersection nodes. After one processor does the coarsening, it will send a bit array to all interested processors describing which nodes of the subset belong the coarse grid. The following calculation of interpolation weights should be done redundantly. On the other hand, a non-blocking coarsening requires identical results in the coarsening of all processors $s\in\sigma^{[i]}$ sharing a block of nodes represented by i. Here, we need an identical numbering of the nodes in $\omega(\sigma^{[i]})$ on all processors. Additionally the coarsening strategy has to be strongly deterministic in that case.

5.4. Parallel coarse grid matrix.

The pattern condition (4.4) for the interpolation matrix $\mathfrak{P} = \{\alpha_{ij}\}_{i\in\omega,\,j\in\omega_C}$ is fulfilled by construction in Alg. 5. Therefore, we can apply Theorem 4.3 with the fine grid matrix K so that the coarse matrix K^H can be calculated fully in parallel, i.e., we can call Alg. 4 locally on each processor s with matrices K_s^H and K_s as appropriate output and input parameters.

5.5. The parallel AMG algorithm.

The parallel AMG algorithm on page 55 simply collects the subroutines of previous sections. The routine PMG-A is the parallel multigrid algorithm with trivial changes.
The coarsening part of Alg. 6 terminates automatically when all sets $\omega(\sigma^{[i]})$ contain only one element.

6. Conclusions.

The main advantage of the proposed parallel AMG algorithms consists in the combination of standard AMG components and standard DD approaches. Therefore, it can be easily implemented in given f.e. packages by combining both parts.

The parallel AMG algorithm is not consistent with respect to the number of processors. i.e., the resulting coarse grids change with the number of processors used. This is no disadvantage in our opinion, because we needed a simple general approach and not one which will change with the hardware.

A distribution of elements on the processors should additionally take into account strongly anisotropic coefficients or similar problems in that (macro-/DD-) coarsening.

Algorithm 6 Parallel AMG PARAMG($\mathsf{K}, \underline{\mathfrak{u}}, \underline{\mathsf{f}}, \omega$)

```
for all $s = 1, \dots, P$ do
    $\mathsf{K}_s^1 \leftarrow \mathsf{K}_s$
    $\omega_s^1 \leftarrow \omega_s$
    $\ell \leftarrow 1$
    repeat
        $\mathfrak{M}_\mathfrak{s} \leftarrow A_s \left( \sum_{q=1}^{P} A_q^T \mathsf{K}_\mathsf{q}^\ell A_q \right) A_s^T$
        $\omega_{C,s} \leftarrow \emptyset$ ,    $\omega_{F,s} \leftarrow \emptyset$
        SIGMACOARSE($\mathfrak{M}_s, \{1, \dots, P\}, \omega_s^\ell, \omega_{C,s}, \omega_{F,s}, \alpha^{\ell+1}$)
        $\mathsf{K}_s^{\ell+1} \leftarrow 0$
        GALERKIN($\mathsf{K}_s^\ell, \mathsf{K}_s^{\ell+1}, \omega_{C,s}, \omega_{F,s}, \alpha^{\ell+1}$)
        $\ell \leftarrow \ell + 1$
        $\omega_s^\ell \leftarrow \omega_{C,s}$
    until termination
end for
PMG-A($\mathsf{K}, \underline{\mathfrak{u}}, \underline{\mathsf{f}}, 1$)
```

REFERENCES

[1] U. GROH, *Local realization of vector operations on parallel computers*, Preprint SPC 94-2[1], TU Chemnitz, 1994. in german.

[2] G. HAASE, *Parallel incomplete cholesky preconditioners based on the non-overlapping data distribution*, Parallel Computing, 24 (1998), pp. 1685–1703.

[3] G. HAASE, B. HEISE, M. JUNG, AND M. KUHN, *FEM⊗ BEM - a parallel solver for linear and nonlinear coupled FE/BE-equations*, DFG-Schwerpunkt "Randelementmethoden", Report 94-16, University Stuttgart, 1994.

[4] G. HAASE, U. LANGER, AND A. MEYER, *The approximate dirichlet decomposition method. part I,II*, Computing, 47 (1991), pp. 137–167.

[5] M. JUNG, *On the parallelization of multi-grid methods using a non-overlapping domain decomposition data structure*, Applied Numerical Mathematics, 23 (1997), pp. 119–137.

[6] K. H. LAW, *A parallel finite element solution method*, Computer and Structures, 23 (1989), pp. 845–858.

[7] A. MEYER, *A parallel preconditioned conjugate gradient method using domain decomposition and inexact solvers on each subdomain*, Computing, 45 (1990), pp. 217–234.

[8] J. W. RUGE AND K. STÜBEN, *Algebraic multigrid (AMG)*, in Multigrid Methods, S. F. McCormick, ed., vol. 3 of Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 1987, pp. 73–130.

[1]http://www.tu-chemnitz.de/ftp-home/pub/Local/mathematik/SPC/spc94_2.ps.gz